```
In[23]:= SAMPLE = 10000;

         Do[DATA[i] = CellularAutomaton[30, RandomInteger[1, 40], 1];
         反復指定       セルオートマトン           乱数整数
          INPUT[i] = Part[DATA[i], 1];
                     部分
          OUTPUT[i] = Part[DATA[i], 2];, {i, 1, SAMPLE}]
                      部分

In[25]:= DATA[3]
         INPUT[3]
         OUTPUT[3]

Out[25]= {{0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1,
           1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0}, {1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0,
           0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1}}

Out[26]= {0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1,
          0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0}

Out[27]= {1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
          1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1}

In[28]:= KARA = {};
         Do[AppendTo[KARA, {INPUT[i]} → OUTPUT[i]];, {i, 1, SAMPLE}]
         … 追加割当て

In[30]:= KARA

Out[30]= {{{0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
             1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1}} →
            {0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1,
            1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},
          {{1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1,
             0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1}} →
            {0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0,
            0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0}, … 9996 …,
          {{1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0,
             1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0}} → {1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0,
            0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0},
          {{0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1,
             0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1}} → {1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1,
            1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0}}

         大きい出力 | 表示を少なく | もっと表示 | すべて表示 | 大きさ制限の設定...

In[31]:= net = NetChain[{GatedRecurrentLayer[100], LinearLayer[40]},
                 ネットワ…   ゲート付き回帰層              線形層
           "Input" → {1, 40}, "Output" → {40}]
            入力を要求

Out[31]= NetChain[  uninitialized    Input                   matrix (size: 1 × 40)
                                  1  GatedRecurrentLayer     matrix (size: 1 × 100)
                                  2  LinearLayer             vector (size: 40)
                                     Output                  vector (size: 40)      ]
```

```
In[32]:=
         training = KARA;
         trained = NetTrain[net, training]
                   ⌊ネットワークの訓練

Out[33]= NetChain[ Input port:        matrix (size: 1×40)
                   Output port:       vector (size: 40)
                   Number of layers:  2                  ]

In[34]:= AAA = CellularAutomaton[30, RandomInteger[1, 40], 1]
               ⌊セルオートマトン          ⌊乱数整数
         Part[AAA, 1]
         ⌊部分
         Part[AAA, 2]
         ⌊部分
         Round@trained[{Part[AAA, 1]}]
         ⌊丸め            ⌊部分
         TrueQ[Round@trained[{Part[AAA, 1]}] == Part[AAA, 2]]
         ⌊真…  ⌊丸め            ⌊部分           ⌊部分

Out[34]= {{1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1,
           1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1}, {0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1,
           1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1}}

Out[35]= {1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0,
          1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1}

Out[36]= {0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1,
          1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1}

Out[37]= {0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1,
          1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1}

Out[38]= True

In[39]:= Do[TESTDATA[i] = CellularAutomaton[30, RandomInteger[1, 40], 1];
         ⌊反復指定         ⌊セルオートマトン          ⌊乱数整数
          TESTINPUT[i] = Part[TESTDATA[i], 1];
                         ⌊部分
          TESTOUTPUT[i] = Part[TESTDATA[i], 2];
                          ⌊部分
          PREDICT[i] = Round@trained[{TESTINPUT[i]}];
                       ⌊丸め
          Print[TrueQ[PREDICT[i] == TESTOUTPUT[i]]]
          ⌊出…  ⌊真かどうか
          , {i, 1, 10}]
```

```
True

True

True

True

True

True

True

True

True

True
```